



## Strumenti per la programmazione

Capitolo 4  
ottobre 2011

### Contenuti

- ◆ Strumenti per la programmazione
  - editing ed editor
  - compilazione e compilatori
  - esecuzione
  - compilatori e interpreti
  - compilazione ed esecuzione di programmi Java
  - Java SE SDK
- ◆ Che cosa fare in pratica
- ◆ Ambienti integrati di sviluppo
- ◆ Eclipse
- ◆ Errori di programmazione

## Strumenti per la programmazione

Per eseguire un programma Java bisogna svolgere le seguenti attività

- editing
- compilazione
- esecuzione

Questo capitolo descrive queste attività e alcuni strumenti che possono essere usati per svolgerle

## Strumenti per la programmazione

```
/* Applicazione che visualizza una frase sullo schermo. */  
public class ScrittoreSulloSchermo {  
    public static void main(String[] args) {  
        System.out.println("ciao a tutti");  
        System.out.println("questo testo introduce");  
        System.out.println("i fondamenti dell'informatica");  
    }  
}
```

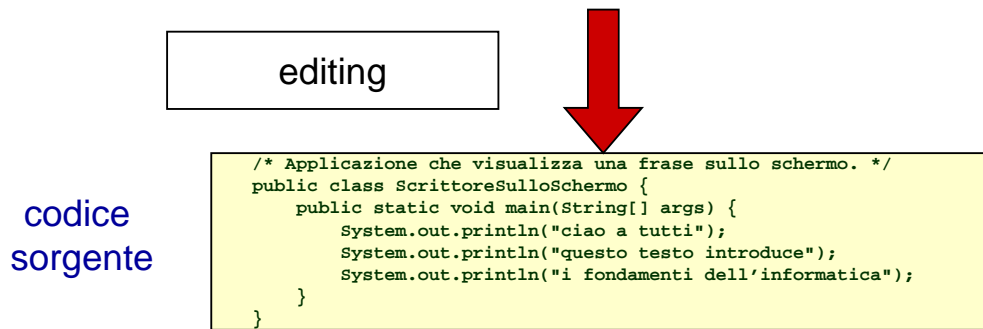
Per eseguire un'applicazione bisogna svolgere tre attività

- **editing**
  - rendere il programma accessibile al calcolatore
- **compilazione**
  - tradurre il programma in un formato eseguibile dal calcolatore
- **esecuzione**
  - far eseguire il programma al calcolatore

## Editing ed editor

Per rendere accessibile al calcolatore una classe o un programma bisogna

- memorizzare la definizione di ciascuna classe in un file di testo
- **editing ed editor**



## Compilazione e compilatori

Un **compilatore** è un'applicazione in grado di tradurre programmi scritti in un linguaggio di programmazione nel linguaggio macchina di un calcolatore

- perché usare un compilatore?

Il linguaggio macchina

- un linguaggio di programmazione
- molto più elementare e primitivo di Java
- specifico di un calcolatore

La traduzione effettuata da un compilatore si chiama **compilazione**

## Linguaggi di programmazione

### Linguaggio macchina

- sequenza di codici binari
- specifico di un calcolatore
- unico linguaggio di programmazione concreto

```
0010110000101100...
```

### Linguaggio assembler

- associa dei codici mnemonici a un linguaggio macchina

```
ADD AX, BX
```

### Linguaggi di alto livello

- il livello di astrazione è alto, perché vicino alla logica della soluzione di problemi

```
delta = b*b - 4*a*c;
```

## Compilazione

### Compilazione

- traduzione (che preserva il significato) da un linguaggio di alto livello a un linguaggio macchina
  - dal **codice sorgente (codice)** – linguaggio di alto livello
  - al **codice eseguibile (eseguibile)** – linguaggio macchina

codice  
sorgente

```
/* Applicazione che visualizza una frase sullo schermo. */  
public class ScrittoreSulloSchermo {  
    public static void main(String[] args) {  
        System.out.println("ciao a tutti");  
        System.out.println("questo testo introduce");  
        System.out.println("i fondamenti dell'informatica");  
    }  
}
```

compilazione

codice eseguibile

```
...  
0010100100010010100111010101001001  
...
```

# Esecuzione

Il codice eseguibile di un programma può venire eseguito direttamente da un calcolatore

- serve il codice eseguibile
- invece, ai fini dell'esecuzione di un programma, il codice sorgente è inutile

codice eseguibile

```
...  
0010100100010010100111010101001001  
...
```

esecuzione

```
ciao a tutti  
questo testo introduce  
i fondamenti dell'informatica
```



# Editing, compilazione ed esecuzione

editing

codice sorgente

```
/* Applicazione che visualizza una frase sullo schermo. */  
public class ScrittoreSulloSchermo {  
    public static void main(String[] args) {  
        System.out.println("ciao a tutti");  
        System.out.println("questo testo introduce");  
        System.out.println("i fondamenti dell'informatica");  
    }  
}
```

compilazione

codice eseguibile

```
...  
0010100100010010100111010101001001  
...
```

esecuzione

```
ciao a tutti  
questo testo introduce  
i fondamenti dell'informatica
```



## Compilatori e interpreti

In realtà, esistono due approcci principali alla traduzione e all'esecuzione dei programmi

- **compilazione**
- **interpretazione**
- i traduttori del primo tipo sono chiamati **compilatori**, quelli del secondo tipo sono chiamati **interpreti**

Un'analogia con la traduzione tra linguaggi diversi

- la compilazione è analoga alla traduzione di un libro
- l'interpretazione è analoga alla traduzione simultanea

## Dipendenza dall'ambiente hardware-software

La compilazione e l'esecuzione sono dipendenti dall'ambiente hardware e software

Un compilatore sa tradurre

- da uno specifico linguaggio di programmazione
  - ad esempio, dal linguaggio C++
- a uno specifico linguaggio macchina
  - relativo a un processore specifico – ad esempio, Intel x86
  - relativo a uno specifico sistema operativo – ad esempio, Microsoft Windows XP/Vista/7

L'eseguibile prodotto da un certo compilatore può essere eseguito solo dai calcolatori corredati di uno specifico ambiente hardware-software

## Compilazione ed esecuzione di programmi Java

Java usa un approccio misto alla compilazione dei programmi

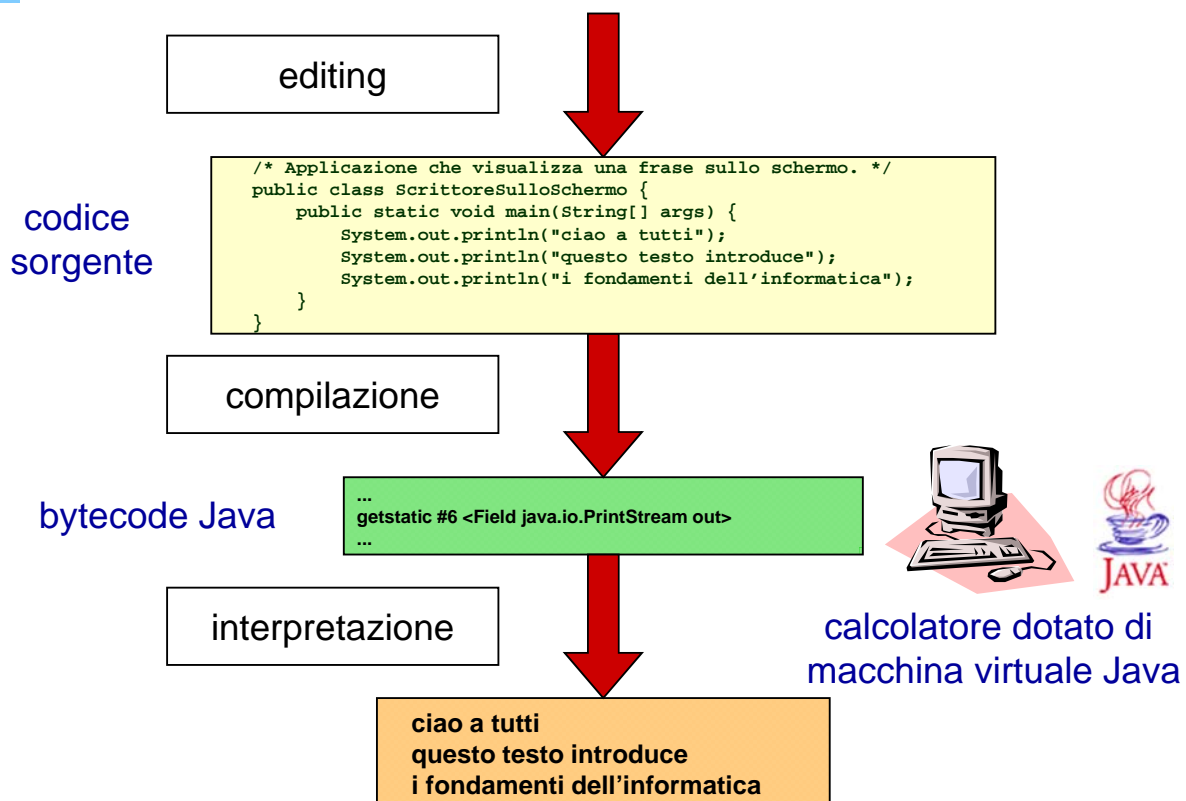
- il codice sorgente Java viene compilato in **bytecode Java**
  - il linguaggio macchina di un calcolatore virtuale
- il programma nella forma di bytecode Java può essere eseguito da un interprete, la **macchina virtuale Java (JVM)**
  - la JVM è un'applicazione che sa eseguire il bytecode Java
  - la JVM rende il calcolatore una macchina virtuale che sa eseguire programmi in bytecode Java

Per ciascun ambiente hardware-software

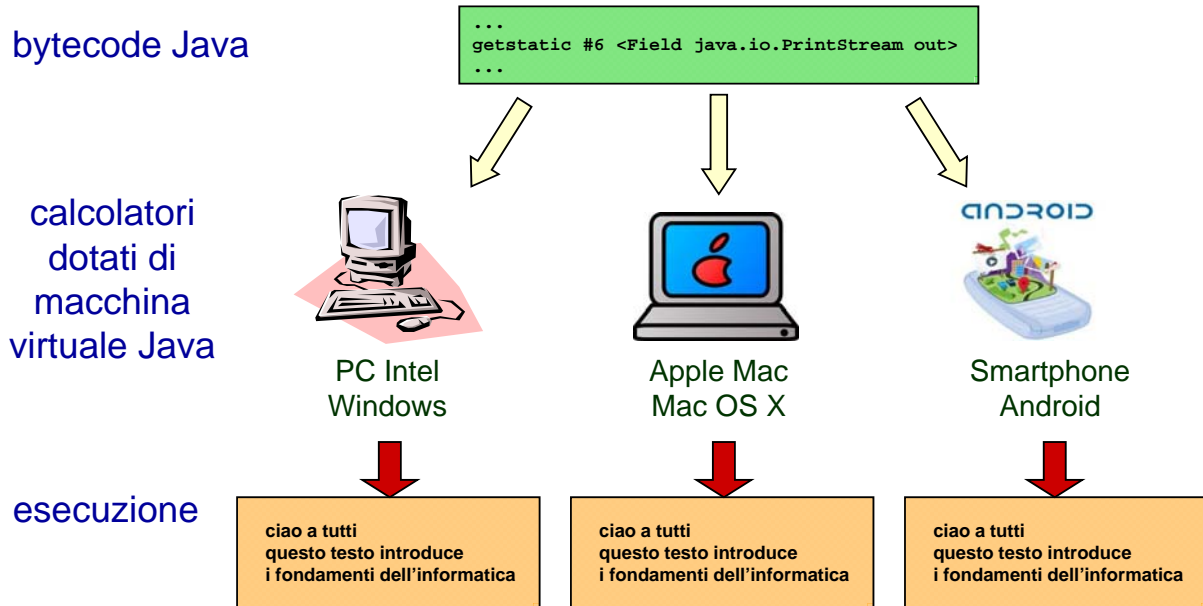
- esiste un diverso compilatore Java
- esiste una diversa macchina virtuale Java

Il bytecode Java prodotto in uno specifico ambiente hardware-software può essere eseguito in qualsiasi altro ambiente hardware-software, purché dotato di una macchina virtuale Java

## Editing, compilazione ed esecuzione di programmi Java



## Write once, run everywhere



## Java Software Development Kit

**Java™ Platform Standard Edition Software Development Kit (Java SE SDK)** è un ambiente di sviluppo per la programmazione in Java

- realizzato dalla Sun Microsystems per diverse piattaforme
- fornisce un certo numero di funzionalità sotto forma di comandi da eseguire in una shell dei comandi
- sito web [java.sun.com/javase/](http://java.sun.com/javase/) – ora [www.oracle.com/technetwork/java/javase/](http://www.oracle.com/technetwork/java/javase/)

Java SE SDK comprende, tra l'altro, i seguenti strumenti di programmazione

- compilatore Java – **javac**
- macchina virtuale Java – **java**
- API (Application Programming Interface) di Java
- applet viewer – **appletviewer**
- debugger – **jdb**
- generatore di documentazione – **javadoc**

## Che cosa fare in pratica

Le tre attività da svolgere nello sviluppo di programmi Java

- editing
  - mediante l'uso di un editor
- compilazione
  - mediante l'uso di un compilatore Java – **javac**
- esecuzione
  - mediante l'uso di una macchina virtuale Java – **java**

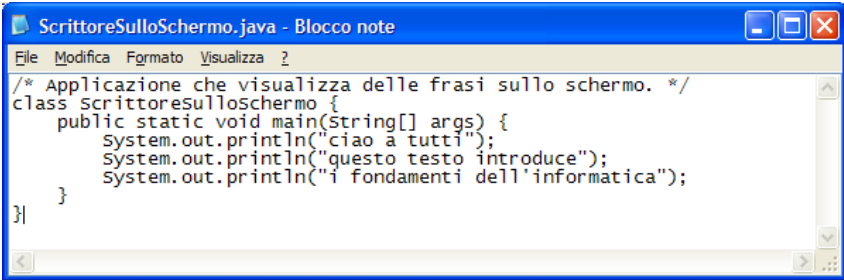
Vengono mostrate due modalità

- uso di alcuni strumenti semplici (primitivi)
- uso di un ambiente integrato di sviluppo

**In quanto segue, si assume che il software di sviluppo necessario sia installato e configurato correttamente nel calcolatore in uso**

## Che cosa fare in pratica – editing

Un editor è **Blocco note** di Windows



```
ScrittoreSulloSchermo.java - Blocco note
File Modifica Formato Visualizza ?
/* Applicazione che visualizza delle frasi sullo schermo. */
class scrittoresulloschermo {
    public static void main(String[] args) {
        system.out.println("ciao a tutti");
        system.out.println("questo testo introduce");
        system.out.println("i fondamenti dell'informatica");
    }
}
```

Il codice sorgente di ciascuna classe deve essere memorizzato in un diverso file di testo

- il file di testo per una classe deve avere come nome il nome della classe seguito dall'estensione **.java**
  - ad esempio, la classe **ScrittoreSulloSchermo** va memorizzata nel file di testo **ScrittoreSulloSchermo.java**
- attenzione a nomi come **ScrittoreSulloSchermo.txt** oppure **ScrittoreSulloSchermo.java.txt**

## Che cosa fare in pratica – compilazione

Il codice sorgente Java deve essere compilato in bytecode Java usando il comando **javac** di Java SE SDK

- ad esempio, per compilare la classe **ScrittoreSulloSchermo** memorizzata nel file **ScrittoreSulloSchermo.java**

```
javac ScrittoreSulloSchermo.java
```

- l'esecuzione di questo comando produce il bytecode di **ScrittoreSulloSchermo** – nel file di nome **ScrittoreSulloSchermo.class**

## Che cosa fare in pratica – esecuzione

Per eseguire una applicazione si deve usare il comando **java** di Java SE SDK

- per eseguire il comando **java** si deve specificare il nome della classe applicazione che si vuole eseguire
- un'applicazione Java viene eseguita come segue
  - viene inizializzata una macchina virtuale Java (JVM)
  - la JVM costruisce l'oggetto classe corrispondente alla classe applicazione
  - la JVM invia il messaggio **main(...)** a questo oggetto classe

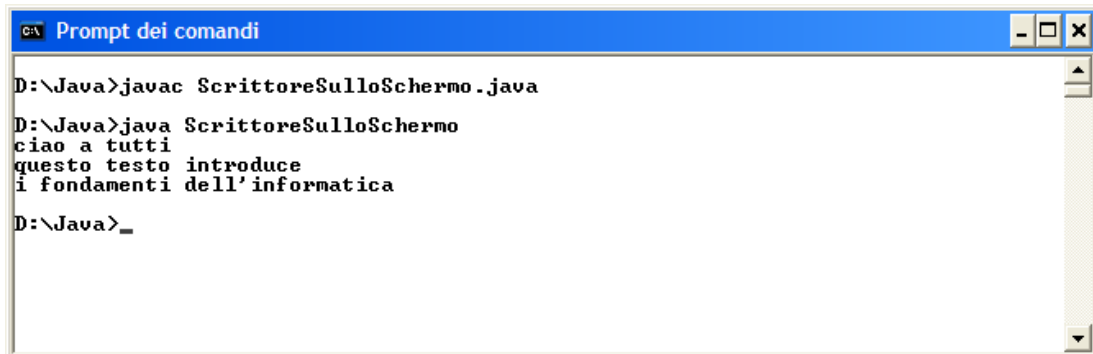
Per eseguire l'applicazione **ScrittoreSulloSchermo** è necessario eseguire il comando

```
java ScrittoreSulloSchermo
```

- si noti che il nome del metodo **main** non compare esplicitamente nel comando

## Che cosa fare in pratica – compilazione ed esecuzione

La seguente figura mostra la compilazione ed esecuzione dell'applicazione **ScrittoreSulloSchermo** nella finestra del prompt dei comandi di Windows



```
C:\ Prompt dei comandi
D:\Java>javac ScrittoreSulloSchermo.java
D:\Java>java ScrittoreSulloSchermo
ciao a tutti
questo testo introduce
i fondamenti dell'informatica
D:\Java>_
```

- **System.out** corrisponde alla finestra in cui viene eseguita l'applicazione

## Ambienti integrati di sviluppo

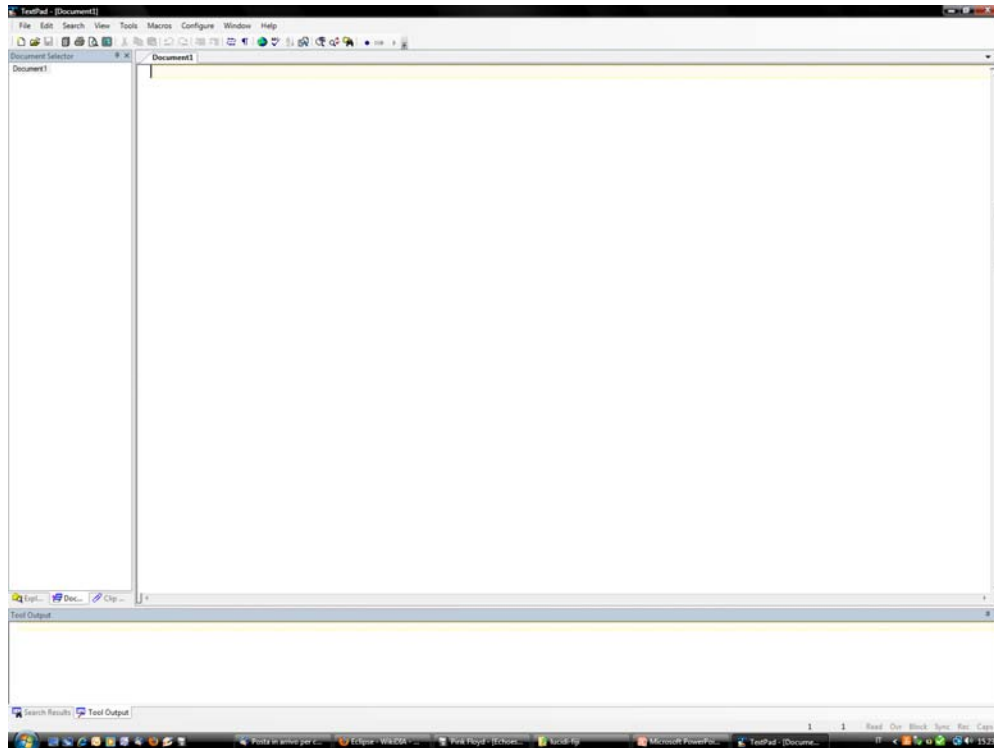
Un **ambiente integrato di sviluppo** (IDE, Integrated Development Environment) è un'applicazione che permette di editare, compilare ed eseguire programmi nell'ambito di un unico ambiente

- TextPad, Eclipse, NetBeans, JBuilder, ...

Funzionalità tipiche

- editor (integrato) guidato dalla sintassi
- accesso al compilatore e alla macchina virtuale Java mediante menu e/o bottoni
- editor visuale di interfacce grafiche (GUI)
- altri strumenti
  - browser della documentazione
  - debugger integrato
  - gestore di progetti e classi

# TextPad

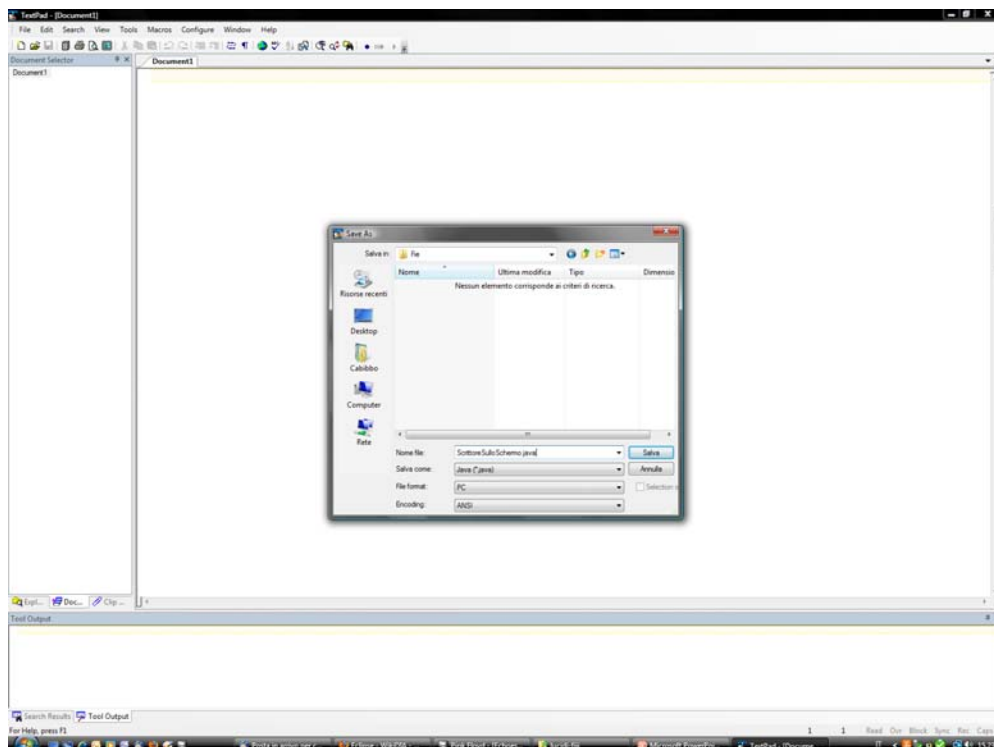


23

Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

# TextPad

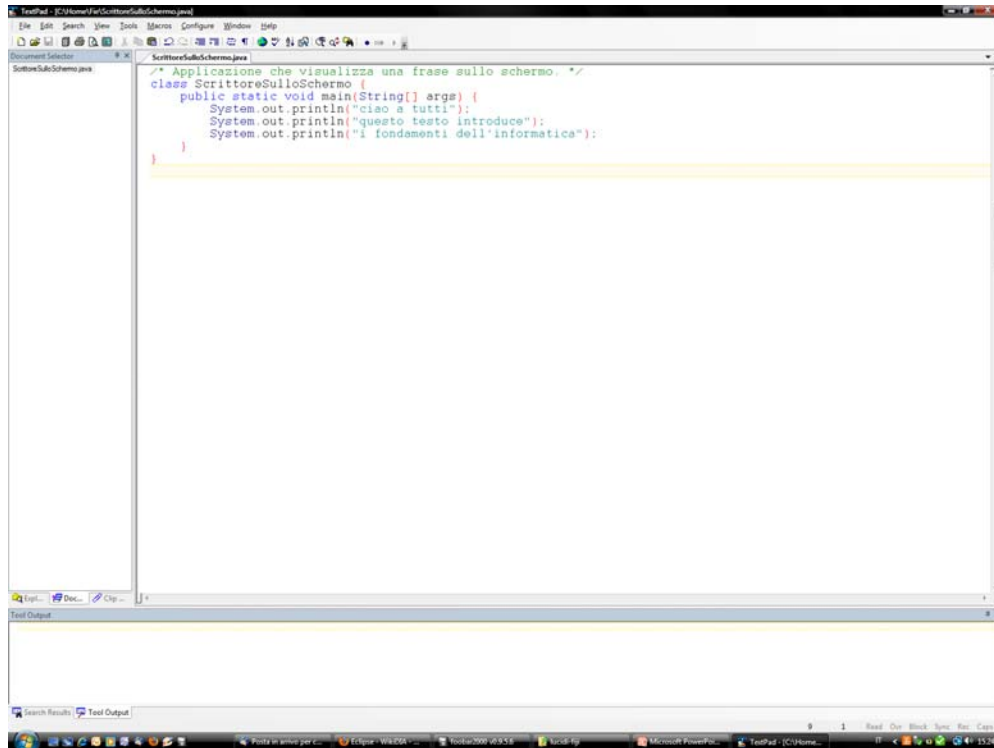


24

Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

# TextPad

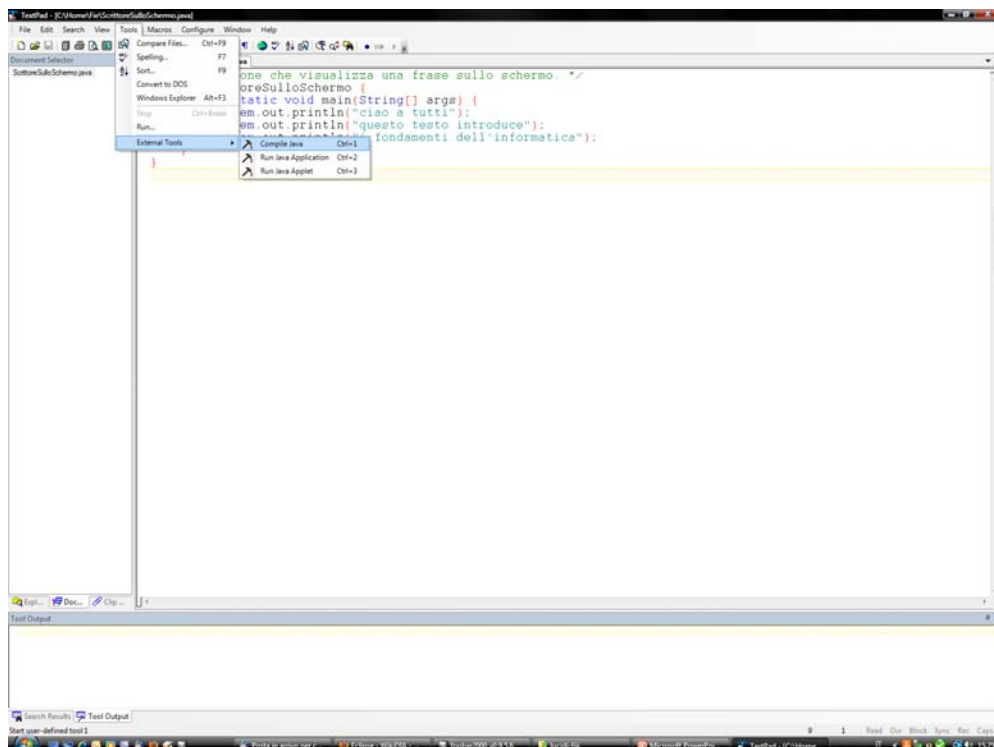


25

Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

# TextPad

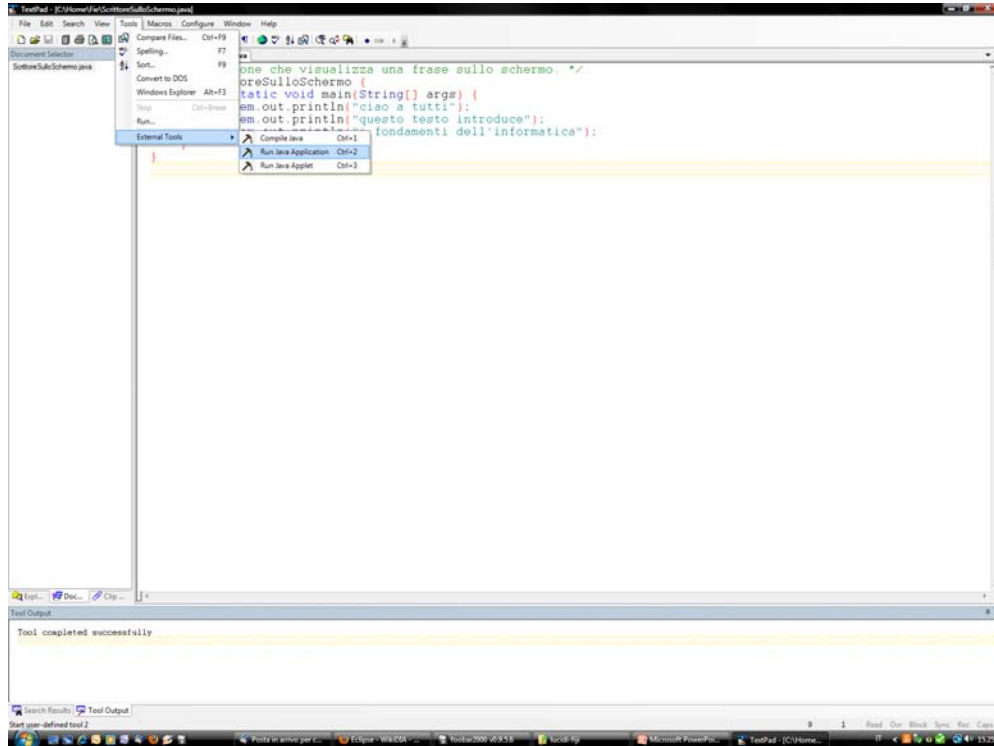


26

Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

# TextPad

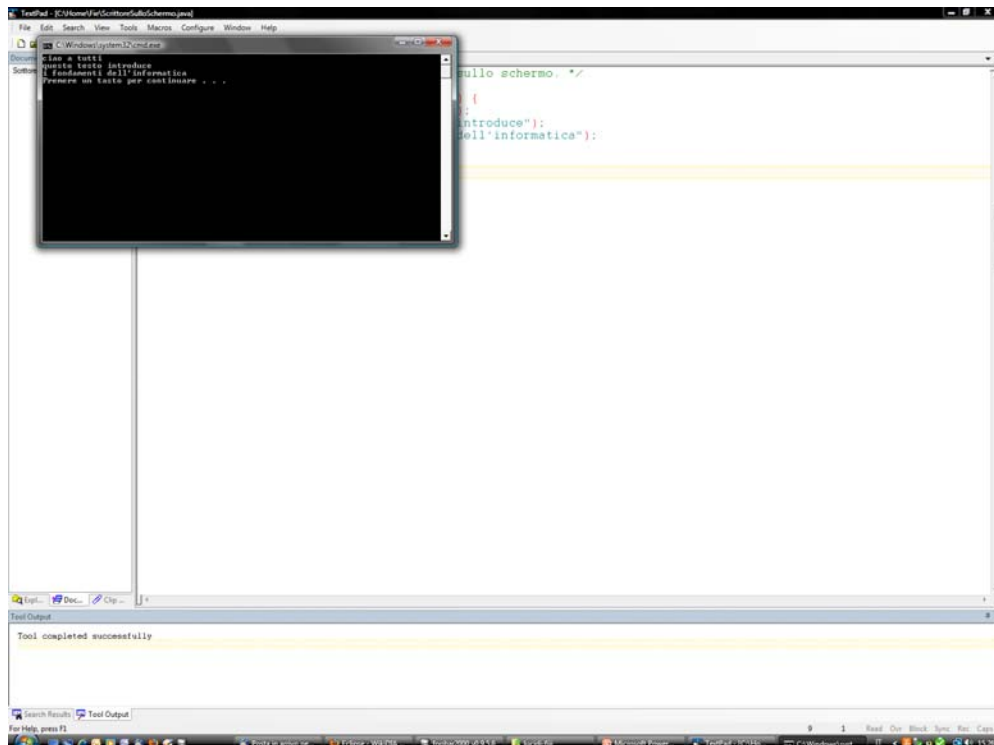


27

Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

# TextPad



28

Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

# Eclipse

**Eclipse** è un ambiente integrato di sviluppo

- inizialmente sviluppato da IBM, poi rilasciato come progetto open-source
- ora è una comunità open-source, il cui obiettivo è lo sviluppo di una piattaforma di sviluppo aperta composta da una famiglia di strumenti estensibili orientati allo sviluppo del software
- sito web [www.eclipse.org](http://www.eclipse.org)
- tante versioni, la più adatta a questo corso è *Eclipse IDE for Java Developers*

L'uso di Eclipse può risultare inizialmente poco immediato

- ma dopo un po' di pratica è molto comodo ed efficace

## Due nozioni di Eclipse

### Workspace

- letteralmente, “spazio di lavoro”
- l'obiettivo di un workspace è di memorizzare tutto il codice relativo ad un grande sistema software
- in pratica, un workspace corrisponde ad una cartella sul file system – e tutte le cartelle ed i file in esso contenuti

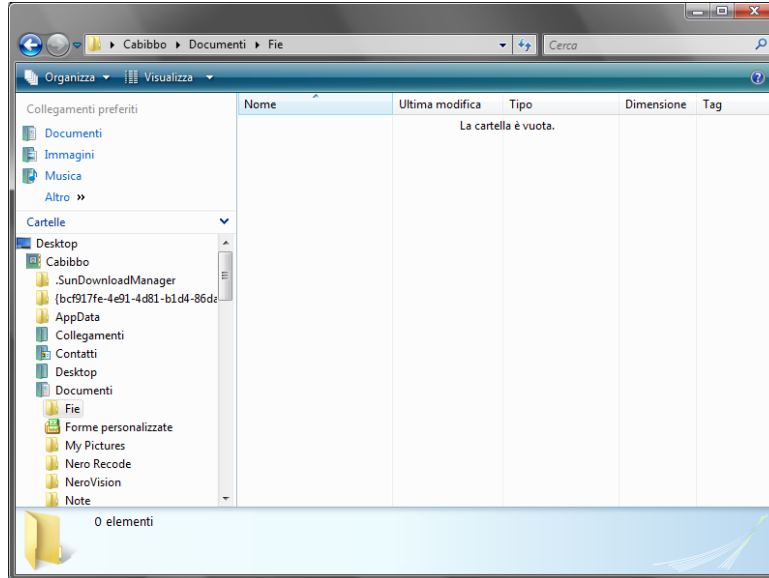
### Project

- letteralmente, “progetto”
- l'idea è che un grande sistema software non è realizzato come un singolo progetto – ma piuttosto come un insieme di progetti correlati
- un project rappresenta dunque una porzione di un sistema software, ovvero una porzione di workspace
- in pratica, un insieme di classi

## Eclipse, in pratica (1)

Per prima cosa, è utile creare sul disco una cartella per memorizzare un workspace

- ad es., si potrebbe dedicare un workspace all'intero corso
  - una cartella **Fie** – nella nostra cartella **Documenti**



31

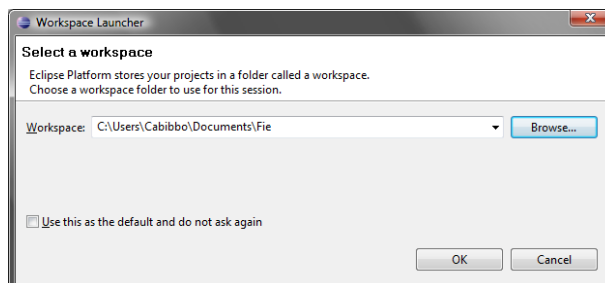
Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

## Eclipse, in pratica (2)

All'avvio di Eclipse viene richiesto di selezionare un workspace

- possiamo dunque selezionare la nostra cartella **Fie** nella nostra cartella **Documenti**



- le volte successive Eclipse si ricorderà della nostra scelta e ce la riproporrà come opzione iniziale

32

Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

## Eclipse, in pratica (3)

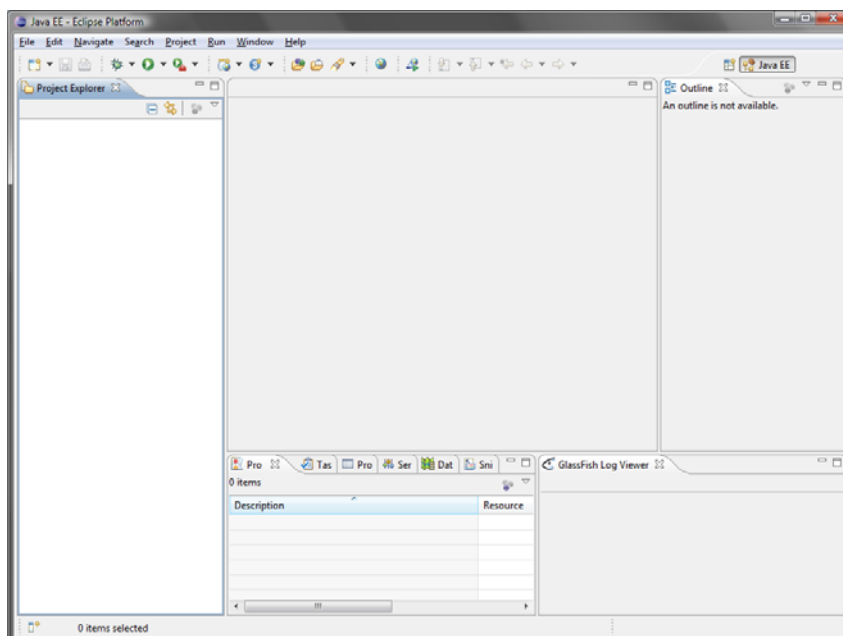
Viene proposta una schermata di benvenuto – “welcome” – che possiamo chiudere

- intervenendo sulla crocetta sulla linguetta, non sulla crocetta per chiudere la finestra



## Eclipse, in pratica (4)

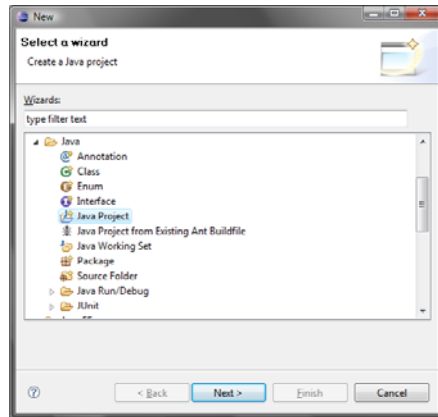
Ora abbiamo di fronte questa schermata



## Eclipse, in pratica (5)

E' ora necessario creare (oppure aprire) un progetto

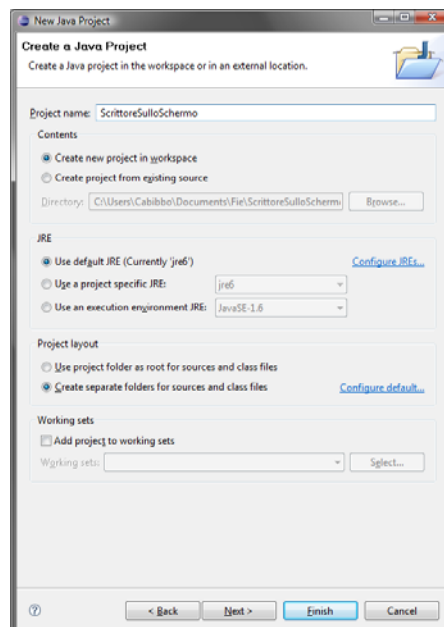
- ad esempio, definendo un progetto per ciascuna applicazione – oppure per ciascun diverso argomento del corso
- dal menu File -> New -> Other..., scegliendo poi Java -> Java Project



## Eclipse, in pratica (6)

Dobbiamo scegliere un nome per il nostro progetto

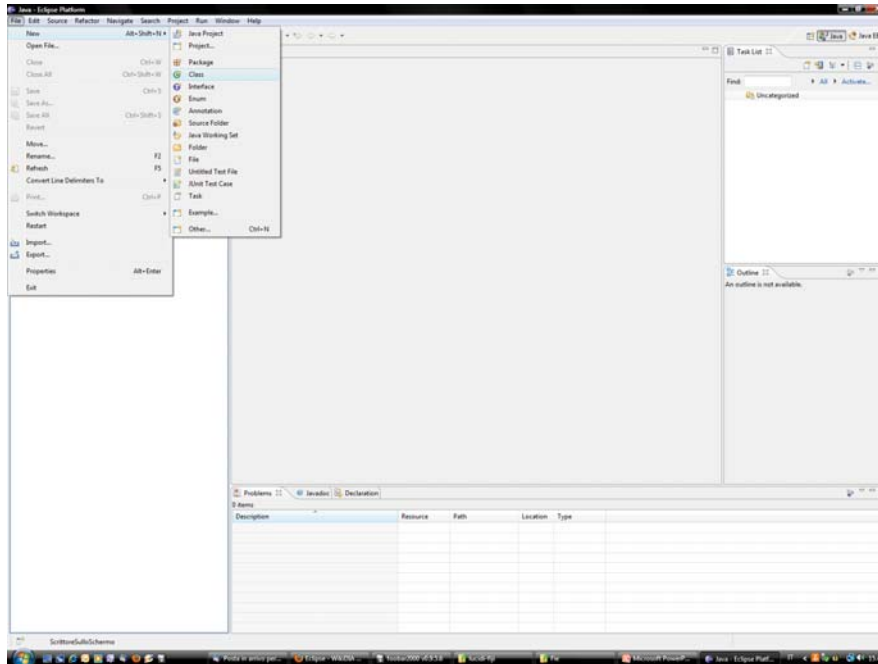
- in un workspace, ogni progetto deve avere un nome distinto
- ad esempio, ScrittoreSulloSchermo



## Eclipse, in pratica (7)

Il prossimo passo è definire una classe

- sempre dal menu File -> New -> Class



37

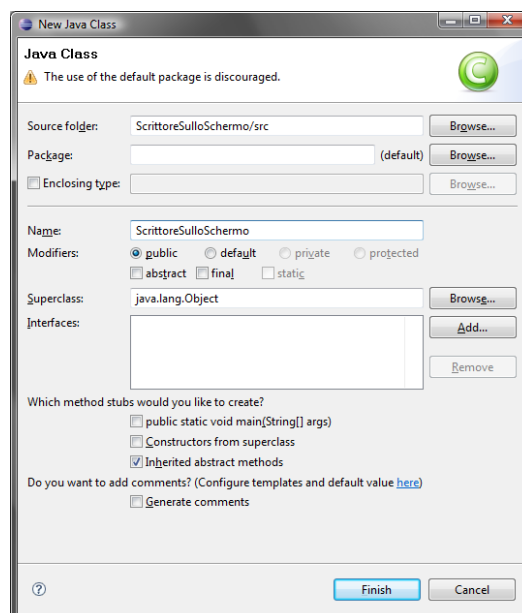
Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

## Eclipse, in pratica (8)

Ogni classe in un progetto deve avere un nome distinto

- ad esempio, ScrittoreSulloSchermo

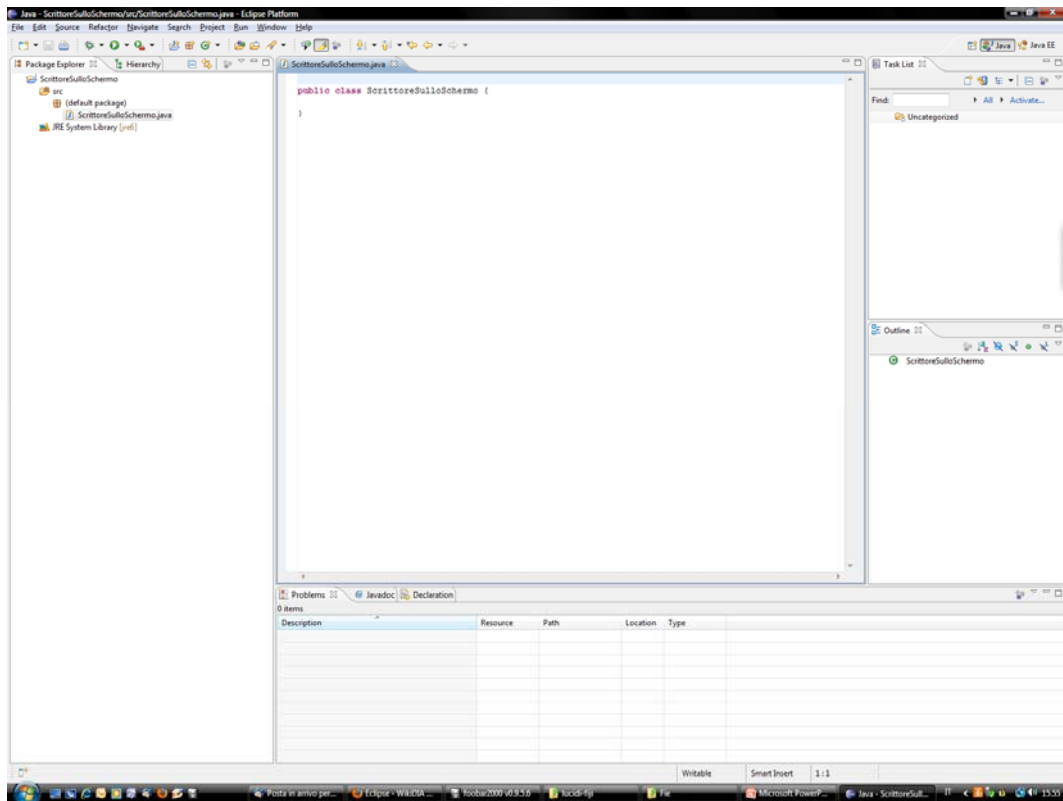


38

Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

## Eclipse, in pratica (9)

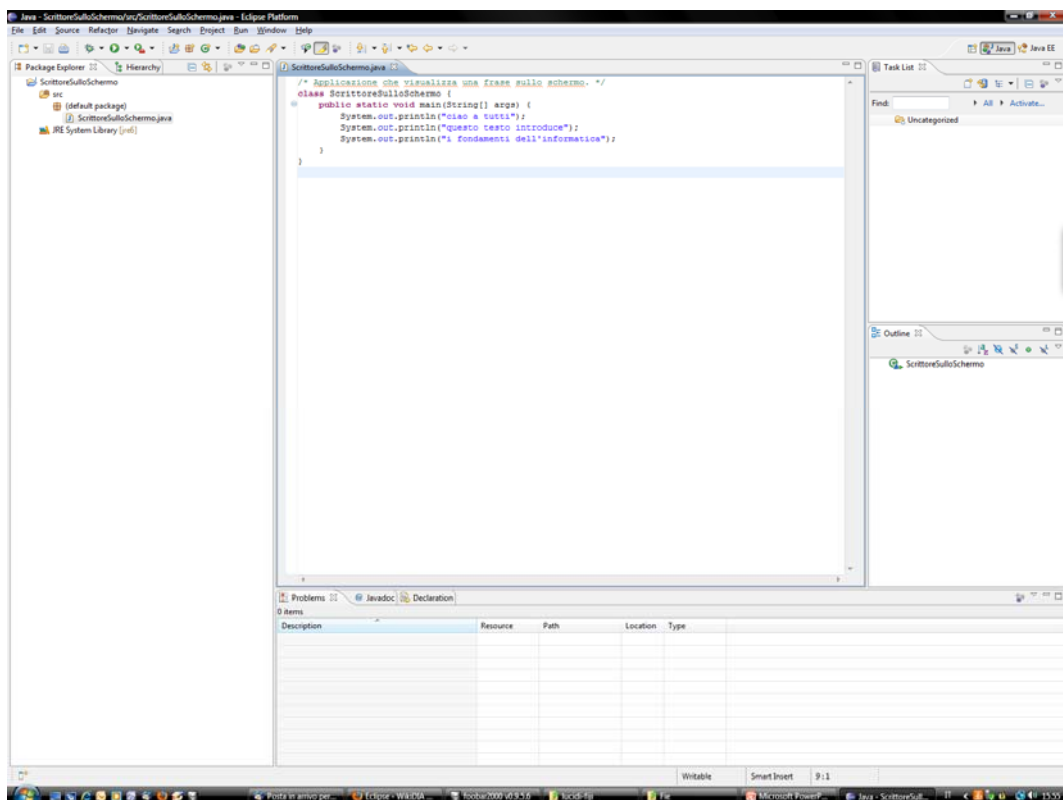


39

Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

## Eclipse, in pratica (10)



40

Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

## Eclipse, in pratica (11)

A differenza di altri IDE, Eclipse prova a compilare – automaticamente e continuamente – tutto il codice che scriviamo, appena lo scriviamo

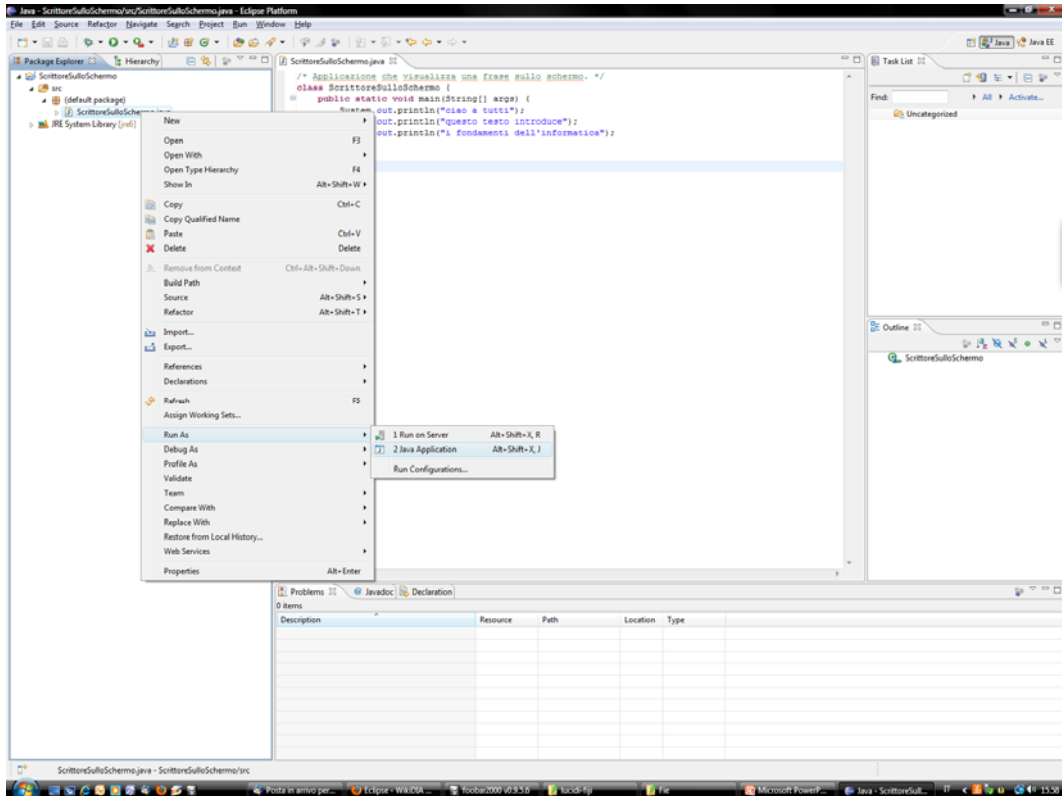
- non dobbiamo compilare il programma in modo esplicito
- ma non è vero che non avviene nessuna compilazione

## Eclipse, in pratica (12)

Per eseguire un programma

- si clicca con il tasto destro del mouse – non il sinistro – sul nome del file per la classe applicazione
- si seleziona la voce Run As -> Java Application
- il risultato dell'esecuzione sarà mostrato nella finestra "Console"
  - in basso a destra

## Eclipse, in pratica (13)

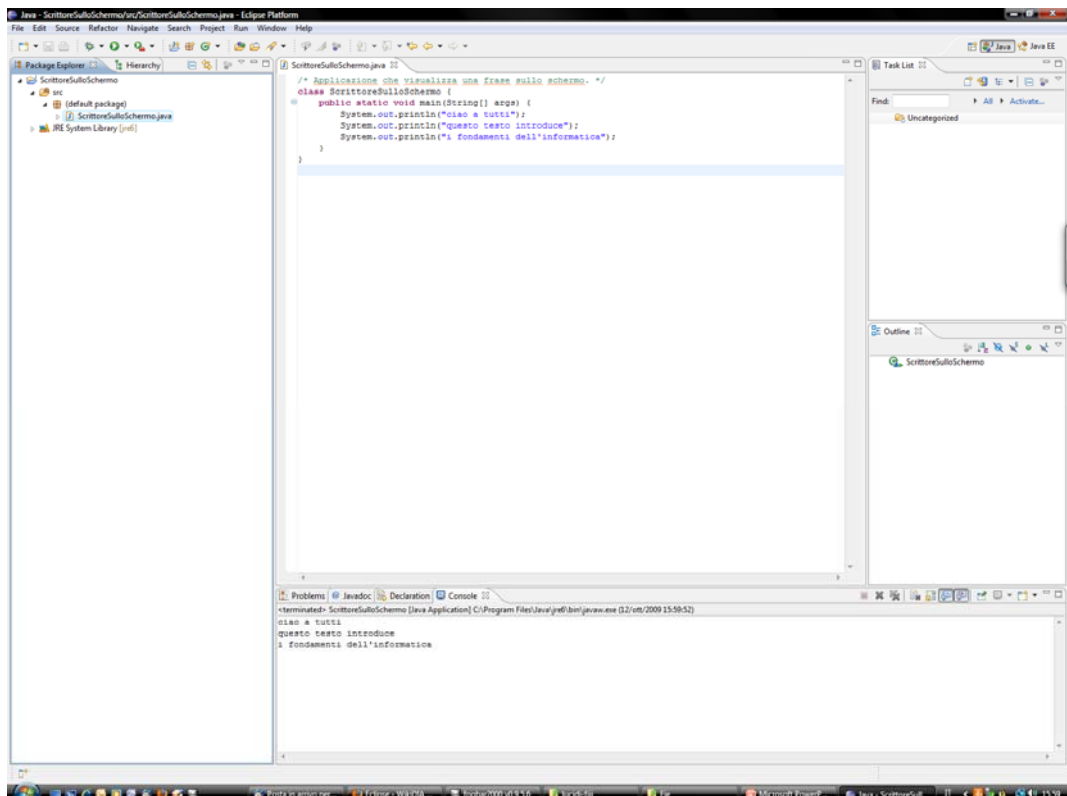


43

Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

## Eclipse, in pratica (14)



44

Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

## Eclipse – in sintesi

### Workspace

- ne devo creare almeno uno
  - uno, una volta per tutte – oppure
  - uno per ciascun sistema da realizzare

### Project

- ne devo creare almeno uno
  - uno per ciascuna argomento del corso – oppure
  - uno per ciascuna applicazione

### Per scrivere un'applicazione

- devo aggiungere – e poi scrivere – tutte le classi che servono
- la loro compilazione è “automatica”
- va richiesta l'esecuzione della “classe applicazione”

## Errori di programmazione

Durante la scrittura dei programmi è possibile commettere degli errori

- **errori di programmazione**

Una classificazione degli errori di programmazione

- *errori grammaticali*
  - ho scritto cose che non hanno significato
- *errori non grammaticali*
  - le cose che ho scritto hanno significato, ma è diverso da quello che dovevo fare

Un'altra classificazione degli errori di programmazione

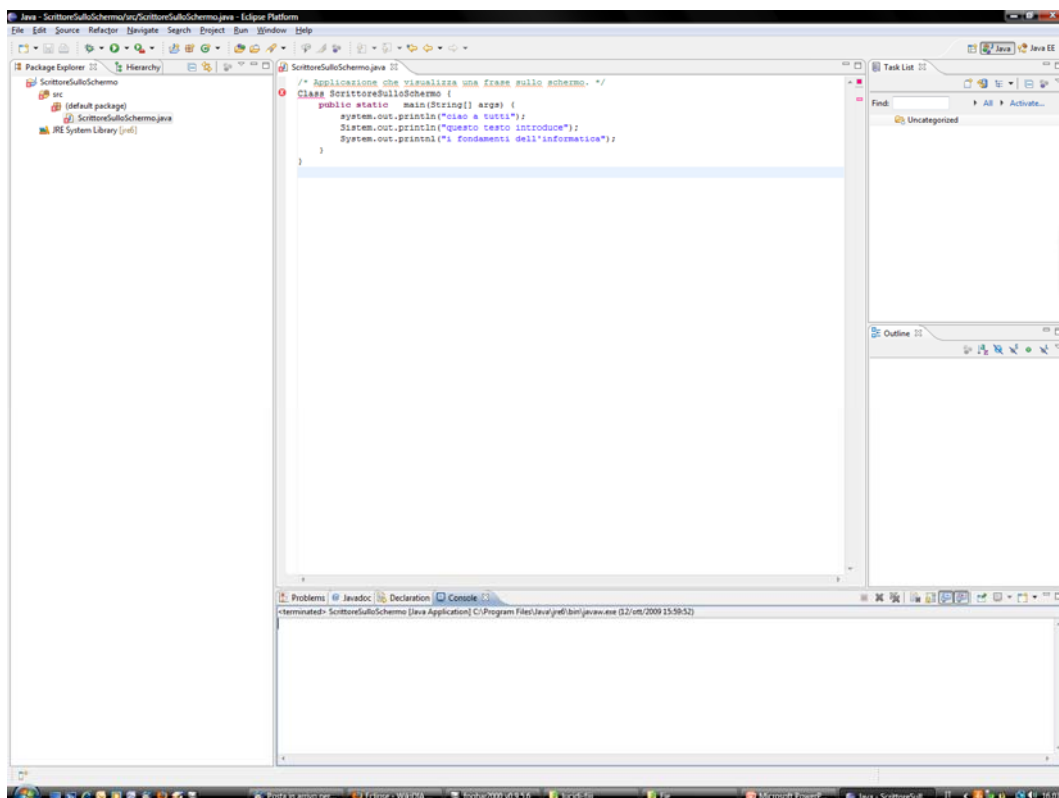
- *errori riconosciuti dal compilatore*
- *errori non riconosciuti dal compilatore*
  - il compilatore non è in grado di riconoscere tutti gli errori!

## Errori riconosciuti e segnalati dal compilatore

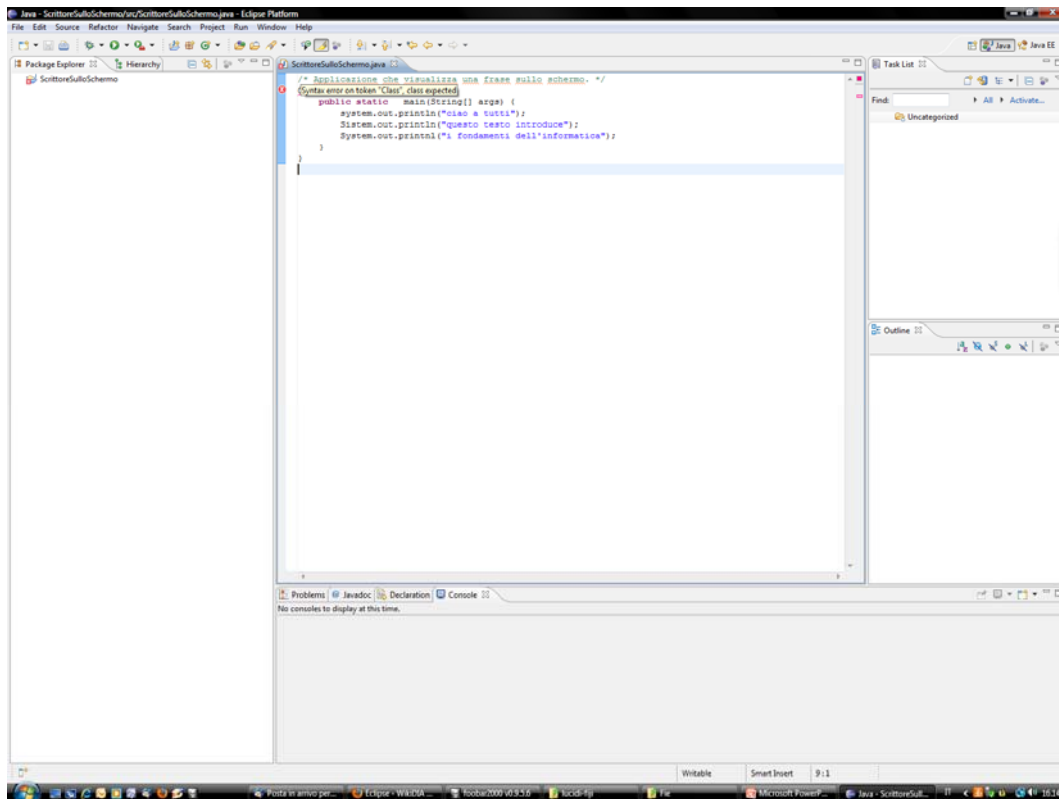
```
/* Applicazione che visualizza una frase sullo schermo. */
public Class ScrittoreSulloSchermo {
    public static main(String[] args) {
        system.out.println("ciao a tutti");
        sistem.out.println("questo testo introduce");
        System.out.println("i fondamenti dell'informatica");
    }
}
```

- è necessario modificare il codice per correggere gli errori segnalati dal compilatore
- spesso sono segnalati uno alla volta

## Errori riconosciuti e segnalati dal compilatore



## Errori riconosciuti e segnalati dal compilatore



49

Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

## Errori non riconosciuti dal compilatore

```
/* Applicazione che visualizza una frase sullo schermo. */  
public class ScrittoreSulloSchermo {  
    public static void Main(String[] args) {  
        System.out.print("ciao a tuti");  
        System.out.println("questo testo introduce");  
        System.out.println("i fondamenti dell'informatica");  
    }  
}
```

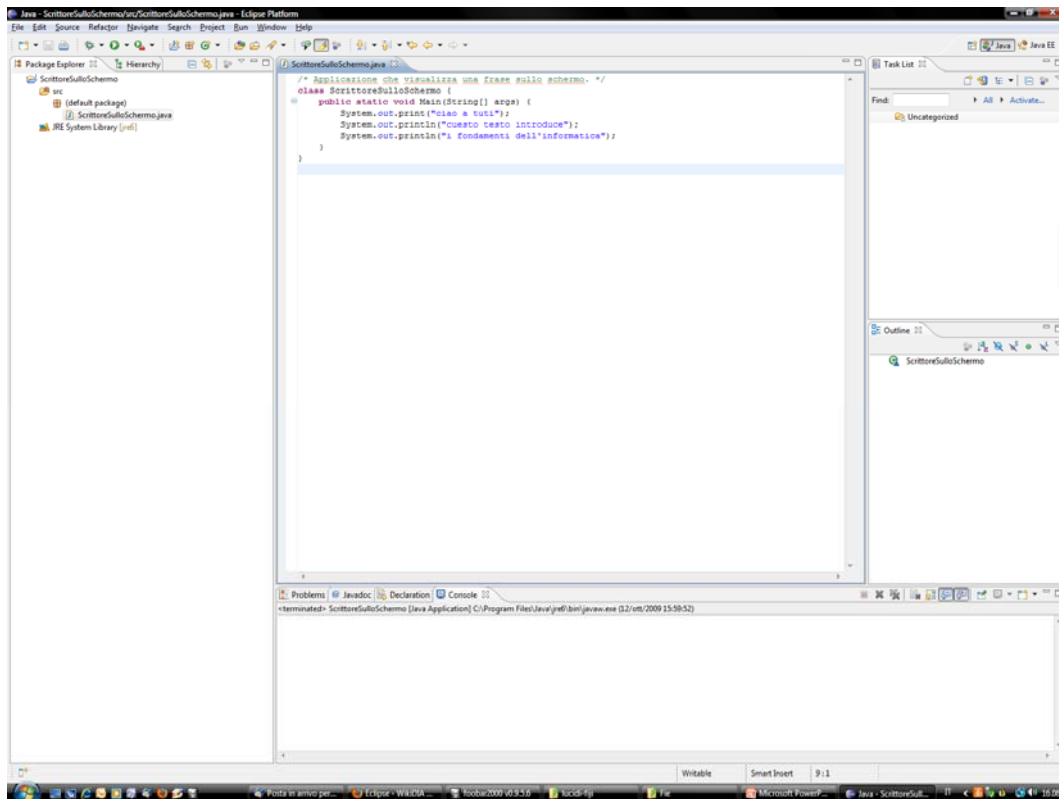
- il compilatore genera il bytecode per questa classe
  - ma non è una classe applicazione
  - se lo fosse, la sua esecuzione produrrebbe un risultato sbagliato

50

Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

# Errori non riconosciuti dal compilatore

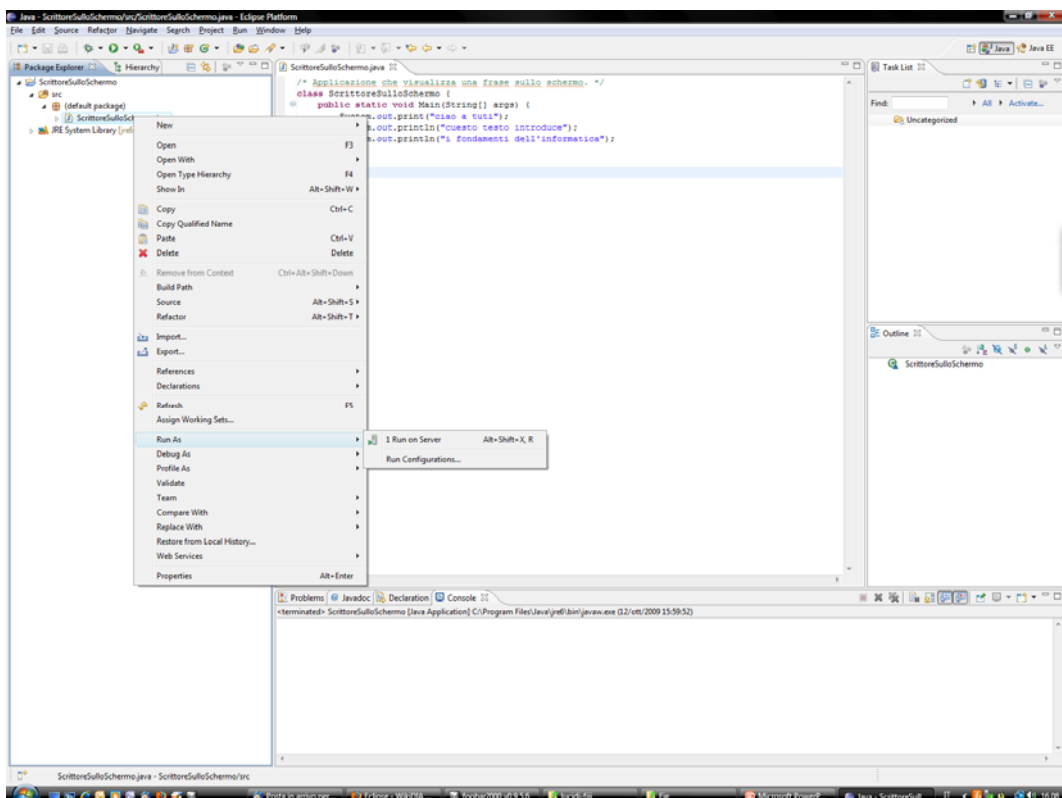


51

Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

# Errori non riconosciuti dal compilatore



52

Strumenti per la programmazione

Fondamenti di informatica: Oggetti e Java  
Luca Cabibbo

## Attività legate alla correzione degli errori

Individuazione degli errori riconosciuti dal compilatore

- per gli errori riconosciuti, il compilatore segnala
  - la posizione dell'errore – file, riga e colonna
  - un messaggio che descrive la tipologia dell'errore
- la comprensione dei messaggi di errore non è sempre facile
  - ad esempio, il compilatore può segnalare un errore diverso (in posizione o tipologia) da quello percepito dal programmatore

Correzione degli errori riconosciuti dal compilatore

Ricerca e correzione degli errori non riconosciuti dal compilatore

- come si fa?

L'individuazione e correzione degli errori

- è un'attività iterativa
- deve essere sostenuta da opportune linee guida di natura sia pragmatica che metodologica